猿小松的进化论

2016-08-25 哈哈小 芳树无人花自落 芳树无人花自落

前言

又到了哈哈小闲聊编程的时间了，是不是有点期待呢，哈哈～

在进入主题之前，请允许我装个c，我们现在讲讲抽象及思想吧，哈哈～

“抽象 - 隐藏系统中不重要的细节” - David Harris

计算机无时无刻不在上演着抽象与具象，一个正真的大牛，是可以在这两则之间随意转换的。要想做到真正的自如转换，绝不是一朝一日所能达成的，需要我们做的是，不断从具体知识之上感受结构的魅力，思想的魅力以获得真正思想上的提高，最终达到解放思想的超然境界。

java作为一门面向对象的语言，以其跨平台性，分布式，可靠安全，多线程，强大的框架支持等特性，常年霸占受欢迎编程语言榜榜首。那么java的魅力究竟在哪里？哈哈小认为，学习任何一门编程语言，比死的知识更重要的，是从具体的一行行代码中体会出来的活的思想。不管是c语言的函数，还是python的模块，亦或是java的封装，这些都有一共性，就是说函数或者说方法吧，他们都能使得代码编写更加逻辑化，效率化，结构化，严谨化，体现的也是编程思想上的一种升华。如果真的能有好的思想上的收获，写出来的代码应该是会给人一种美感的。

哈哈小有一篇文章谈到java学习之路，最后的阶段，当你对具体的东西都足够了解之后，要学习的就是框架了。其实，在我们写代码的所有阶段，都有框架的概念，只不过是不同阶段，由于不同的认识深度，框架有优劣之别而已。可以说，抽象即框架，好的框架是以具体为前提而构建的，框架之下，需要的是具体的填充。所以，知识点的熟练掌握也很重要哦，哈哈～。

下面，哈哈小就以一小段故事让大家领会一下java的魅力，体会一下抽象之美，感受一下新手与大牛所写代码的截然不同以及认识到代码结构（或者说设计模型）的重要性。

正文

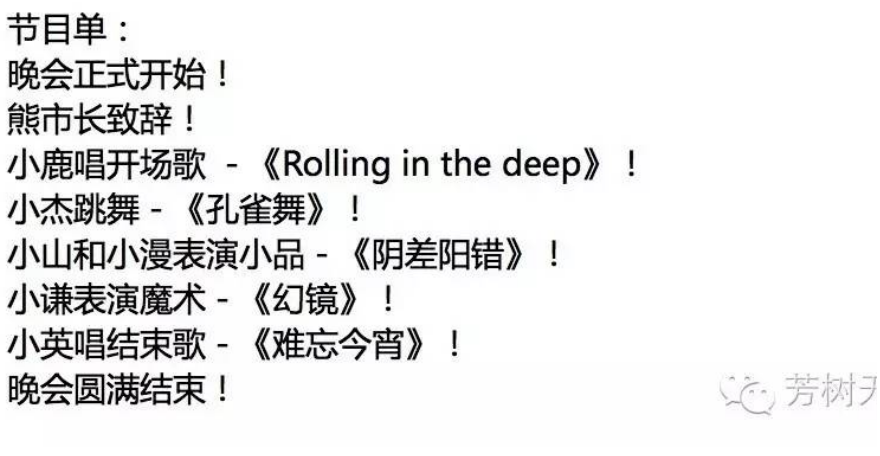
动物们通过一年的辛勤劳作终于迎来了大丰收，为表感激，熊市长邀请了时下最受欢迎的表演艺术家，计划在乔伊动物城中心广场举办一次盛大的庆祝晚会，届时将欢迎乔伊动物城的所有动物进城观看。

晚会上，将会有开场唱，跳舞，相声，魔术，结束歌五个节目依次登台演出。受邀请的表演者包括鹿妈妈的儿子 - 歌唱家小鹿，当红小鲜肉 - 孔雀小杰，年度最佳相声搭档 - 鹦鹉小山和鹦鹉小漫，百变魔术师 - 变色龙小谦，乔伊动物城著名歌唱家 - 黄鹂小英。

在晚会筹备阶段，为使节目编排的更有序，熊市长请来了程序设计师 - 猿小松，叫他写一段简单的程序，使得晚会节目单一目了然。

接收到熊市长的任务后，猿小松用java语言通宵达旦把任务赶了出来。以下是猿小松程序实现的具体过程。

按照熊市长的要求，最后程序要求输出的结果应该如下：

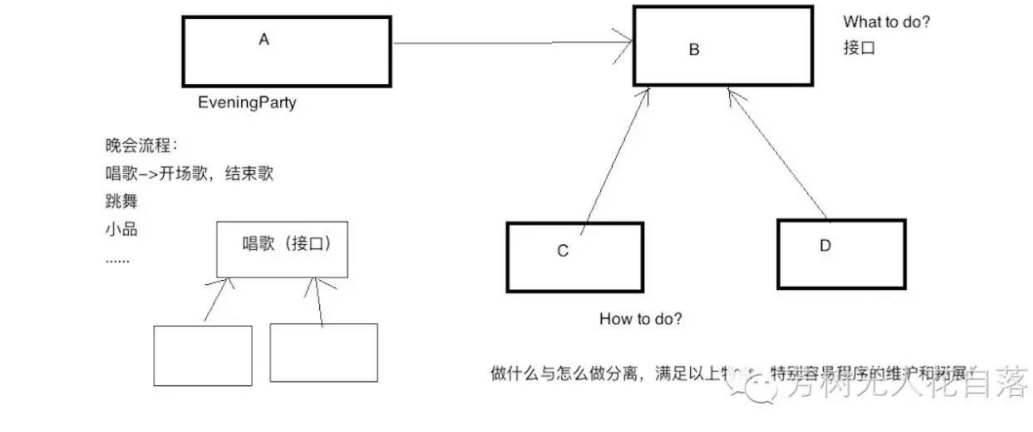


一开始，猿小松花了不到五分钟的时间，根据要求迅速把程序写出来了。觉得这任务太简单了，不用加班，轻轻松松搞定，暗自窃喜。

他写的最初的程序是这样的：

```java

public class EveningParty {
    public static void main(String[] args) {
        new EveningParty().execute(); // to start the evening party
    }

    public void execute() {
        // party arrangement
        // beginning song -> dancing->opusculum ->magic->ending song --- what
        // does the program do?
        System.out.println("晚会正式开始！");
        System.out.println("熊市长致辞！");
        System.out.println("小鹿 唱开场歌 －《Rolling in the deep》！");
        System.out.println("小杰 跳舞－《孔雀舞》！");
        System.out.println("小山和小漫 表演小品－《阴差阳错》！");
        System.out.println("小谦表演魔术－《幻镜》！");
        System.out.println("小英唱结束歌－《难忘今宵》！");
        System.out.println("晚会圆满结束！");

    }
}

```


片刻， 猿小松转念一想，像这样一类的问题以前也遇到过，是“做什么”与“怎么做”的问题。如果程序写到这就结束了，那将会是一个非常不利于维护和拓展的低劣的程序。猿小松想到了java作为一门面向对象的语言，具有继承，多台，封装三大特性。经过一番思考，猿小松有了如下的结果。




```java
package v2;

public interface Singable {
    void Beginningsing();

    void Endingsing();
}
```



```java
package v2;

public class XiaoLu implements Singable {

    @Override
    public void Beginningsing() {
        System.out.println("小鹿 唱开场歌 —《Rolling in the deep》！");

    }

    @Override
    public void Endingsing() {
        System.out.println("小鹿唱结束歌—《难忘今宵》！");

    }

}
```


他创建了四个接口分别去定义唱歌，跳舞，小品，魔术的方法，同时创建了以表演者名字命名的类并让这些类分别去实现对应接口中的方法（上图是小鹿实现Singable接口中方法的展示）。通过这样的修改，猿小松做到了“做什么”与“怎么做“的分离。

除此，主程序也修改成了如下所示：

```java
package v2;

public class EveningParty {
    public static void main(String[] args) {
        new EveningParty().execute(); // to start the evening party
    }

    public void execute() {
        // party arrangement
        // beginning song -> dancing->opusculum ->magic->ending song --- what
        // does the program do?
        System.out.println("晚会正式开始！");
        System.out.println("熊市长致辞！");

        Singable singable1 = new XiaoLu();// 准备 歌曲节目
        singable1.Beginningsing();// 开场歌

        Danceable danceable = new XiaoJie(); // 准备舞蹈
        danceable.dance();

        Performable performable = new XiaoShanAndXiaoMan(); // 准备小品节目
        performable.perform();

        Magic magic = new XiaoQian();// 准备魔术
        magic.doMagic();

        Singable singable2 = new XiaoYing();// 准备 歌曲节目
        singable2.Endingsing();// 结束歌

        System.out.println("晚会圆满结束！");

    }
}
```


这时，猿小松又想到一个问题，要是小鹿因为需要照顾生病鹿妈妈或者别的什么原因而无法出席晚会演出，最后只能让承当结束歌部分的小英同时承担小鹿开场歌部分的工作。这样的话，小松就不得不在上图所展示的类中的第十五行稍作修改。但是小松觉得这样的情况如果多次发生，每次都要到主程序里去修改，很是不妥，于是猿小松对自己的程序又做了一番修改。

小松具体是这样做的，为了不想修改晚会类，引入了工厂结构（中介）以达到将“做什么”与“怎么做”进行解耦合的目的。

工厂类是这样写的：

```
1  package v3;
2
3  public class Factory {
4
5      // 提供准备开场歌手的方法
6      public static Singable getSingable1() {
7          return new XiaoLu();
8
9      }
10
11     // 提供准备结束歌手的方法
12     public static Singable getSingable2() {
13         return new XiaoYing();
14
15     }
16
17     // 提供准备舞蹈的方法
18     public static Danceable getDanceable() {
19         return new XiaoJie();
20
21     }
22
23     // 提供准备小品的方法
24     public static Performable getPerformable(
25         return new XiaoShanAndXiaoMan();
26
27     }
28
29     // 提供准备魔术的方法
30     public static Magic getMagic() {
31         return new XiaoQian();
32
33     }
```

晚会类15行到28行修改成了如下所示：

```
12          System.out.println("晚会正式开始！ ");
13          System.out.println("熊市长致辞！ ");
14
15          Singable singable1 = Factory.getSingable1();// 准备 歌曲节目
16          singable1.Beginningsing();// 开场歌
17
18          Danceable danceable = Factory.getDanceable(); // 准备舞蹈
19          danceable.dance();
20
21          Performable performable = Factory.getPerformable(); // 准备小品节目
22          performable.perform();
23
24          Magic magic = Factory.getMagic();// 准备魔术
25          magic.doMagic();
26
27          Singable singable2 = Factory.getSingable2();// 准备 歌曲节目
28          singable2.Endingsing();// 结束歌
29
30          System.out.println("晚会圆满结束！ ");
31
```

猿小松这会儿想着，程序写到这应该没有什么可挑剔的了吧，于是合上电脑，翘首望去，看到悬于墙上的时钟已经指向了十二点。时间也不早了，猿小松想着如往常一样，闭目凝神一番便作休息。就在这时，猿小松又想到了一种可以使程序更加完善一下的策略。方正已经这么晚了，无妨多花点时间来写个漂亮码，猿小松心里是这样想着。静下心来，一会儿，键盘的嘀嗒嘀嗒声又回响在了整个房间。

猿小松最后对程序完善使用的是反射技术，将演员实例类写入配置文件。猿小松，创建了一个配置文件party.properties。

项目结构和文件具体内容如下：


```
Singable1 = v4.XiaoLu
Singable2 = v4.XiaoYing
Danceable = v4.XiaoJie
Performable = v4.XiaoShanAndXi
Magic = v4.XiaoQian
```

```java
package v4;

import java.util.ResourceBundle;

public class Factory {
    // 提供准备 开场歌手方法
    public static Singable getSingable1() {
        // 读取配置文件 src/party.properties
        String className = ResourceBundle.getBundle("party").getString("Singable1");

        // 已知完整类名 获得对象
        try {
            Object obj = Class.forName(className).newInstance();
            return (Singable) obj;
        } catch (Exception e) {
            e.printStackTrace();
            throw new RuntimeException("因故未能出席！");
        }
    }

    // 提供准备 结束歌手方法
    public static Singable getSingable2() {
        // 读取配置文件 src/party.properties
        String className = ResourceBundle.getBundle("party").getString("Singable2");

        // 已知完整类名 获得对象
        try {
            Object obj = Class.forName(className).newInstance();
            return (Singable) obj;
        } catch (Exception e) {
            e.printStackTrace();
            throw new RuntimeException("因故未能出席！");
        }
```


利用反射技术对**Factory**类做了大修改（上图展示只是**Factory**中的两个方法，其他的方法仿照修改）

这样的话，整个程序的可维护性和可拓展性又到达了另一个层次。如果小鹿因故不能出席，小英代之，只需要将配置文件中第一行的XiaoLu改成XiaoYing即可。或者说，如果熊市长能邀约到另一名当红小生 - 云雀华，去取代小鹿的工作，这样的话，只需要做两件事，一是配置文件XiaoLu改成YunQueHua， 二是创建YunQueHua类去实现Singlable接口中的Beginningsing方法即可。

猿小松通过对程序不断的提升和完善，最后终于达到了自己满意的状态。次日，熊市长对猿小松的最后成果甚为满意，并对猿小松这种精益求精的态度给予了极大的肯定。

三天后的晚上，同城的几乎所有的动物齐聚一堂，共赏晚会，其乐融融。当时场面盛大，节目精彩纷呈，声浪迭起，掌声雷动。按照晚会流程，经过近两个小时的演出，晚会在欢快的氛围中顺利圆满结束，猿小松当心的问题当然也没有出现。归途中的动物们，有的为着精彩的晚会而感到意犹未尽，有的为着一年的收成而满心欢喜，有的因怀着对来年的美好期许而心花怒放。

未完待续

结束语

这里要强调的是，程序 = 数据结构 + 算法，不管是任何的一种算法，还是数据结构，没有最完善，只有更完善和永无止境的探

索和最求！

声明

本文为原创文章，文章仅代表编者个人观点，如需转载请注明作者信息及本文链接，谢谢尊重个人劳动成果。

长按指纹“识别二维码” 快速关注

点击阅读原文

阅读原文 投诉